ELSEVIER
PII: S0954-1810(97)10003-6
Artificial Intelligence in Engineering **12** (1998) 261–273

0954-1810/98/$19.00

Operator guidance system for coal washing

Hernan Villanueva[a] **& Harmeet Lamba**[b]

[a]*Department of Electrical and Computer Engineering, The University of Newcastle, Callaghan, NSW 2308, Australia*
[b]*BHP Research, Newcastle Laboratories, PO Box 188, Wallsend, NSW 2287, Australia*

(Received 25 July 1996; revised version received 20 July 1997; accepted 28 July 1997)

In this paper we present the analysis, design and implementation of the Ash Control Model (AshMod), a real-time knowledge-based operator guidance system (OGS) in the coal washing domain. The complexity of coal washing is such that a knowledge modelling methodology was needed for the development of this operator guidance system. The Knowledge Acquisition and Design Structuring (KADS) methodology guided us in the development of these knowledge models. We present an integration of the KADS methodology with the G2 real time implementation environment.

AshMod assists the operator in monitoring the plant, performing fault diagnosis, and in plant optimisation. Furthermore, it assists the operator in maximising clean coal yield while keeping ash (impurity) content within acceptable limits. AshMod performs deep reasoning through the use of KADS-inspired knowledge models that capture purpose, function, structure, behaviour and heuristics. Knowledge validation and maintenance are facilitated through the use of graphical object-oriented knowledge models, implemented in G2.

AshMod has been developed for the B and C coal washing plants operated by Broken Hill Proprietary Limited (BHP) at Port Kembla, Australia.[1,2] Owing to our focus on generality and reuse during the development of this operator guidance system, we expect that much of AshMod can be reused in future OGS developments in BHP-operated coal washeries, sinter plants, blast furnaces, coke ovens etc. AshMod is currently undergoing online testing at the coal washing plants.

Key words: operator guidance system, KADS, coal washing.

1 COAL WASHING

The Flat Products Division (FPD) of BHP Steel operates the B and C plant coal washeries. The essential purpose of the coal washery is to improve the quality of raw coal by removing impurities.[4,5] The two plants process 250 and 350 tons of raw coal per hour, respectively. Unwashed coal is fed into the washery to reduce the ash (impurity) content in the coal. The coking coai and energy coal output from the washery is used for export and for internal use. High ash content decreases the value of coal. Hence the plant operation is geared towards controlled the ash content in the washed coal.

Sizing screens are used to classify the coal on the basis of size. The different sizes of coal are then washed by separate processes optimised to remove impurities from coal of that size.[4,5] Jigs are used to wash large coal, cyclones to wash small coal, and flotation cells are used to wash fine coal (Fig. 1). The jigs and cyclones are gravity stratification processes that exploit the fact that all removable impurities associated with coal are higher in specific gravity than the coal. Flotation cells, on the other hand, exploit the hydrophobic nature of coal to separate it from impurities.

Operators continuously monitor the plant by looking at large volumes of sensor data. Sensors operating in a rough industrial environment sometimes record incorrect readings. The operators validate sensor readings to determine whether the measurements can be relied upon. The readings are subject to noise, which the operators filter out. For safety and cost considerations, sensors are difficult to calibrate. Operators perform mental adjustments to sensor readings to account for calibration offsets. They follow statistical process control (SPC) guidelines to isolate plant components that are out of control. Operators recognise process trends and perform fault detection by using associations between trend patterns and faults. They use knowledge about cause-and-effect relationships to diagnose the root cause of detected faults. Operators have knowledge

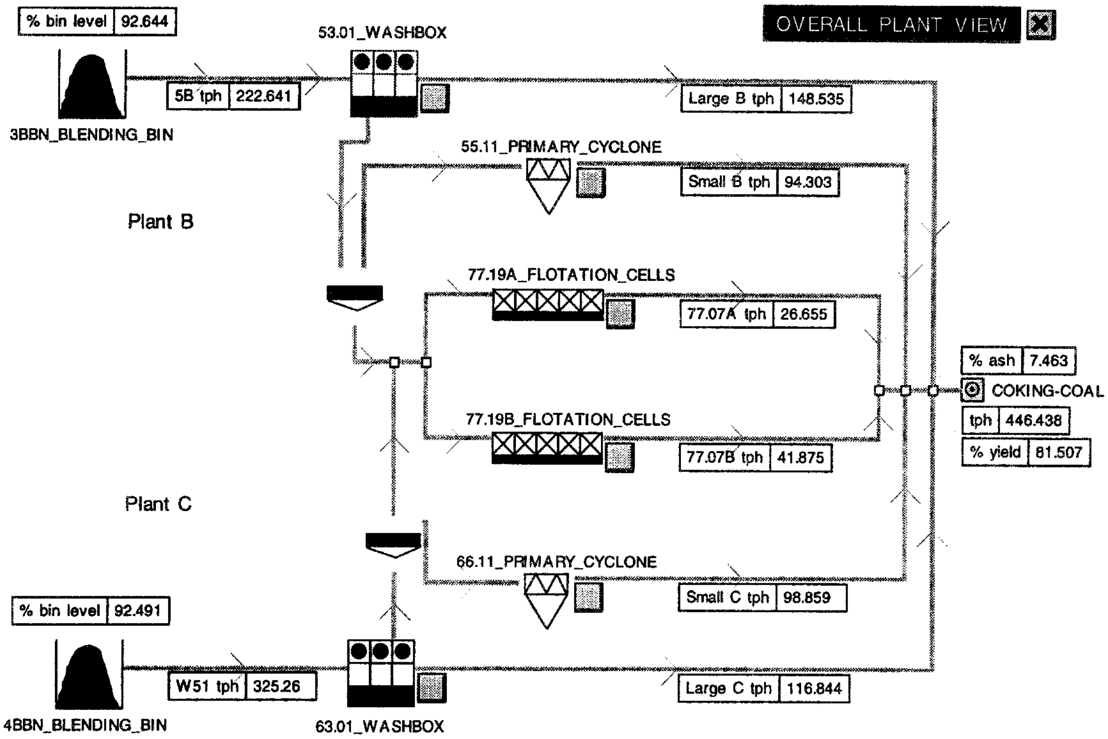


Fig. 1. Simplified schematic of the B and C plant coal washery.

about the action needed to correct an identified malfunction. They identify opportunities for fine tuning and optimising the process.

Hence the tasks performed by operators in the coal washery may be classified into three major categories: monitoring, diagnosis and optimisation. To be an effective decision support system, an operator guidance system (OGS) must perform these tasks in real time, providing timely and reliable recommendations to the operator.

2 KADS

The acquisition of knowledge and its appropriate representation, validation and maintenance is a challenging task in complex industrial domains. Traditional rule-based artificial intelligence (AI) approaches have provided limited methodological support for knowledge acquisition, representation, validation and maintenance. Hence, for the development of this OGS, the need was felt for a systematic methodology that could support the complete knowledge engineering lifecycle. This support was provided by the Knowledge Acquisition and Design Structuring (KADS) methodology.

The aim of KADS is to fill the need for a structured methodology for knowledge-based system (KBS) projects by constructing knowledge models.[6,7] The methodology was developed over the past decade under the European Esprit programme. When the KADS project was originally conceived, sometime in 1983, little interest in methodological issues existed in the AI community. The prevailing paradigm for building KBSs was rapid prototyping. Since then, developers of KBSs have come to realise that a methodology for developing KBSs is just as necessary as in conventional software engineering.

However, conventional software engineering methodologies tend to be inadequate for knowledge engineering exercises. Knowledge acquisition, the most time-consuming and unstructured aspect of knowledge engineering, is not fully supported by conventional methodologies.

Many knowledge engineering methods, on the other hand, are heavily oriented towards prototyping. It can be argued that prototypes often develop into finished systems with poor or no design and a lot of patchwork. This can make such systems difficult to extend and maintain. Heuristics and prototypes can capture only shallow knowledge since no attempt is made to model or understand the process. An unstructured list of rules can take a long time to search, making real-time operation difficult. Unexpected interactions between rules can make a knowledge base very hard to maintain.

Deep knowledge is captured by modelling the objectives, geography, connectivity, hierarchy and physics of all the components of the plant. Models result in a deeper understanding and hence a greater scope for intelligent behaviour. The natural modularity assists in pruning rule searches, minimises the side effects between rules, and makes knowledge maintenance much simpler.

The domain models presented in Sections 5–7, and the task models detailed in Sections 8–11, were motivated by the KADS methodology. The KADS domain models capture knowledge about the purpose and function (Section 5), structure (Section 6), behaviour and heuristics (Section 7) associated with the plant and its components.

The KADS task models capture knowledge about how individual tasks such as monitoring (Section 9), diagnosis (Section 10) and optimisation (Section 11) are performed. A more in-depth treatment of these domain and task models is provided elsewhere,[14] together with the role played by KADS in their development.

3 G2

G2 is an application development environment for building and deploying intelligent applications.[8,9] Numerous such applications are currently in operation in industrial environments. G2 supports the application lifecycle with:

- object-oriented tools, graphical tools and a structured natural language that encourage close communication between developers and end-users;
- object libraries and functional modules that promote incremental development;
- concurrent, multi-user access in distributed, client-server development and run-time environments;
- full interactive capabilities over remote networks;
- re-use of application objects and modules in future applications;
- rule-based and model-based reasoning;
- interfacing capabilities with external programs, files and databases;
- a suite of integrated tools that boosts development productivity.

G2-based applications have been developed for the purpose of:

- analysis and diagnosis;
- quality management;
- plant-wide optimisation;
- dynamic scheduling;
- decision support.

Modelling techniques allow domain expertise to be effectively captured, represented, applied and maintained. A user-friendly interface allows the potential for on-site knowledge maintenance[12,13] by the domain experts.

Real-time rules, procedures and models can look for trends before problems become big and expensive. Costly disruptions can be minimised by quickly diagnosing problems and taking corrective actions.

By combining expert knowledge with analytical methods, an optimisation strategy can be developed that aims at maximising yield, quality and throughput.

Being object oriented, G2 is well suited for a knowledge modelling approach.

4 ASHMOD

Operator knowledge is lost when an operator retires. This knowledge loss is prevented by capturing it in an operator guidance system. Due to differences in levels of expertise, operators introduce quality variability into the process. By providing consistent recommendations, an operator guidance system reduces this variability, hence improving product quality. During process upsets, operators have to digest large volumes of sensory data and take prompt corrective action. An operator guidance system[11,15] acts as an assistant, thus reducing the cognitive overload. Fault-related shutdowns result in loss of production. Such shutdowns can be prevented by the timely identification of process upsets by an operator guidance system. An operator guidance system can help continuously improve and optimise a process, thus increasing product quantity and quality.

Our operator guidance system, AshMod, helps plant operators in maximising the clean coal yield while keeping ash (impurity) content within acceptable limits.[3] It contains knowledge for interpretation of process data, and is intended to advise operators on process conditions and control actions. AshMod enhances the performance of the plant by allowing the operator to make more informed decisions. By providing consistent and timely guidance to the operator, AshMod assists an operator in getting higher yield whle maintaining consistent product quality. It is not intended to replace the operator, but rather to act as an assistant. Throughout the development of this operator guidance system, plant operators have been actively involved in the acquisition and validation of domain knowldge and in user interface specifications. This has made operator acceptance easier and has hence contributed to enhanced system useability and correctness.

AshMod applies the knowledge modelling principles propounded by the KADS methodology and is implemented using G2. It assists the operator in the three major task categories identified earlier: monitoring, diagnosis and optimisation. By automating the monitoring task, AshMod reduces the cognitive load on the operator. AshMod encapsulates diagnostic knowledge and helps the operator to perform quick and consistent fault detection, diagnosis and removal. AshMod also asssts the operator in the optimisation task by providing appropriate perfective recommendations.

This is a soft real-time system. The data inputs to AshMod change every 4 min. The system response time is of the order of 1 s (the time it takes AshMod to provide an appropriate recommendation to the operator), while the operator response time is of the order of 1 min (the time it takes for the operator to verify a recommendation and act on it). Since coal washing is a relatively slow process, it may take up to a quarter of an hour before an operator action has an observable impact on the process variables.

An overview of AshMod's functionality can be provided by means of an example. For each plant component, AshMod monitors a set of process indicators. By applying statistical process control (SPC) criteria on these indicators, AshMod is able to determine whether a plant component is operating in a normal state. SPC rules compare the values and trends of the process indicators over a specified time window, against the process mean and against upper and lower control limits.

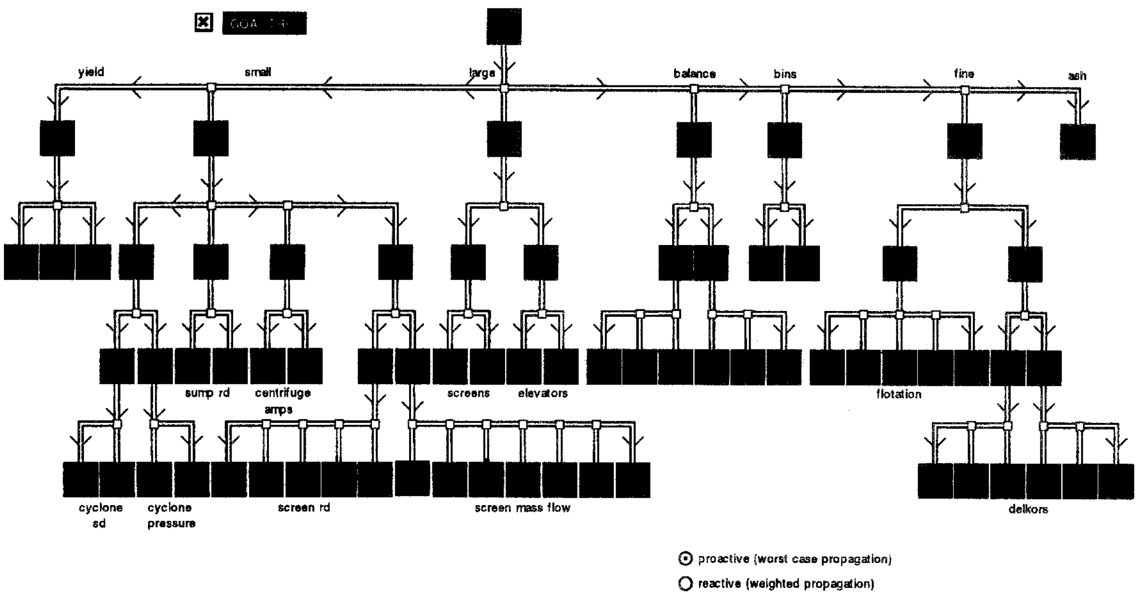


Fig. 2. The AshMod GTST workspace.

The process component icons in the schematic workspace, as well as the goals in the Goal Tree–Success Tree associated with this component, are highlighted in red when the component fails to satisfy the SPC criteria. AshMod performs trends analysis on the process indicators associated with the suspected components, to identify process faults. Each process variable has a long-term trend (over a four-hour window), a medium trend (over a one-hour window) and a short-term trend (over a 30 min window). These trends may be increasing, decreasing or steady. The discovery of a fault is based on the observation of a specified combination of trends.

The fault cause workspace is displayed and the identified faults are highlighted in red. For every identified symptom there are many faults and malfunctions that could have been the cause. Prior probabilities are used to identify the most likely cause, which is highlighted in blue. Messages are sent to the message workspace to inform the operator about the suspected components, the identified faults and the potential faults and malfunctions.

The operator is kept in the loop by asking him to accept or reject an AshMod recommendation. An identified fault is one that AshMod is able to recognise by observing the trends of relevant process indicators, or one that has been explicitly identified by the operator. A potential fault is a fault hypothesis constructed by AshMod. It is a plausible fault, but there is inadequate evidence to support it. These are presented to the operator, together with an indication of AshMod's level of confidence in the likelihood of the fault's occurrence.

When the operator accepts a potential fault it becomes an identified fault, and AshMod examines its possible causes. This process continues till a list of possible (hypothesised) malfunctions is generated together with AshMod's level of confidence. Finally, the operator accepts a malfunction message to indicate that he agrees that the malfunction was the root cause of the identified fault(s). AshMod then displays the recommended action that the operator needs to perform to correct the malfunction. Once the corrective action has been taken, the inertia associated with the process often results in a delay of up to half an hour before the effect of the action becomes visible through the process indicators. During this period, AshMod assumes that the malfunction has been corrected, and ensures that the same root cause is not again generated as a possible (hypothesised) malfunction. This time delay is recorded as a knowledge attribute inside the malfunction object, together with the last time this malfunction was corrected. Thus AshMod utilises an algorithm similar to one used by cache memory systems to determine when to re-enable the malfunction.

For example, the medium density problem object, S5 (Fig. 6) contains 30 min as the action reaction time attribute. High cyclone ash may be caused by S5. AshMod records the time, t_1, when the operator acknowledges medium density as the cause of high cyclone ash. When S5 is

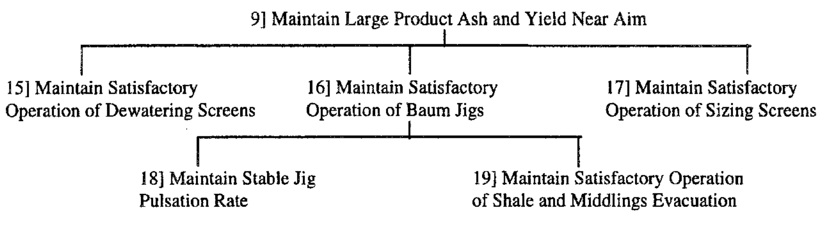


Fig. 3. The 'Large coal circuit' subtree from the AshMod GTST.

G3, a goal	
Notes	OK
Item configuration	none
Names	G3
Goal statement	"Optimal Operation of Small Coal Circuit"
Goal weight	0.2
Goal value	0.0
Circuit	small
Trend	smallb

Fig. 4. Small coal circuit goal G3.

again identified as a possible cause at time t_2, then AshMod checks to see if $t_2 - t_1 > 30$ min to determine whether it is legitimate to reconsider S5 as a cause.

If all plant components satisfy the SPC criteria there is no need for fault diagnosis, and AshMod turns to optimisation. The plant optimisation workspace is displayed. Here, the current operating point of each plant circuit is compared with the target. The circuit operating point is determined by its yield (the ratio of product to feed tons per hour) and product ash level. Each circuit has a target ash and yield level determined on the basis of current feed quality (i.e. the amount of impurity in the raw coal being fed into the plant). For each circuit, the 'distance' of the current ash and yield levels from the target determines the circuit's 'optimality coefficient'. The circuit with the largest 'optimality coefficient' is the one that is furthest away from the target operating point and offers maximum scope for improvement. Hence this circuit is highlighted in red on the optimisation workspace, and the appropriate perfective action needed to adjust the circuit setpoint is displayed to the operator.

To perform its tasks intelligently, AshMod needs static domain knowledge that describes its operating environment, as well as dynamic task knowledge that describes how these tasks are to be performed. In the following sections we describe the domain models that capture knowledge about the purpose and function of plant components, the plant structure, neuristic fault–cause associations, and the behaviour of plant components. We then describe the task models that capture knowledge associated with the three major tasks performed by AshMod: monitoring, diagnosis and optimisation.

5 GOAL TREE–SUCCESS TREE

The Goal Tree–Success Tree (GTST)[10] is a KADS domain knowledge model that captures the purpose and function of the plant and its various components. It is a kind of means–end model in which the plant's intentional aspects are captured through a problem reduction strategy. The GTST is used to represent the mental model used by superintendents and operators in operating a complex plant. The knowledge is modelled by a tree structure of hierarchically related goals and subgoals that must be satisfied for the correct operation of the plant.[3] The upper section of the GTST (goal tree or GT) consists of goals and subgoals that capture prupose. The lower section (success tree or ST) consists of the success criteria that captures function (Figs 2 and 3).

The construction of the goal tree begins with the identification of the essential objective of the entire plant. In AshMod, this top goal is the 'safe, environmentally sound, and cost effective operation of the plant'. The top goal is then divided into subgoals that are necessary and sufficient for its achievement. Looking downward from a supergoal, one can see how it is satisfied through the satisfaction of its subgoals. Conversely, looking upward from a subgoal, one can see why it is needed for the satisfaction of the supergoal. The verification of the lowest level goals is based on plant instrument readings (raw or filtered), operator activities, events, or some calculated parameters.

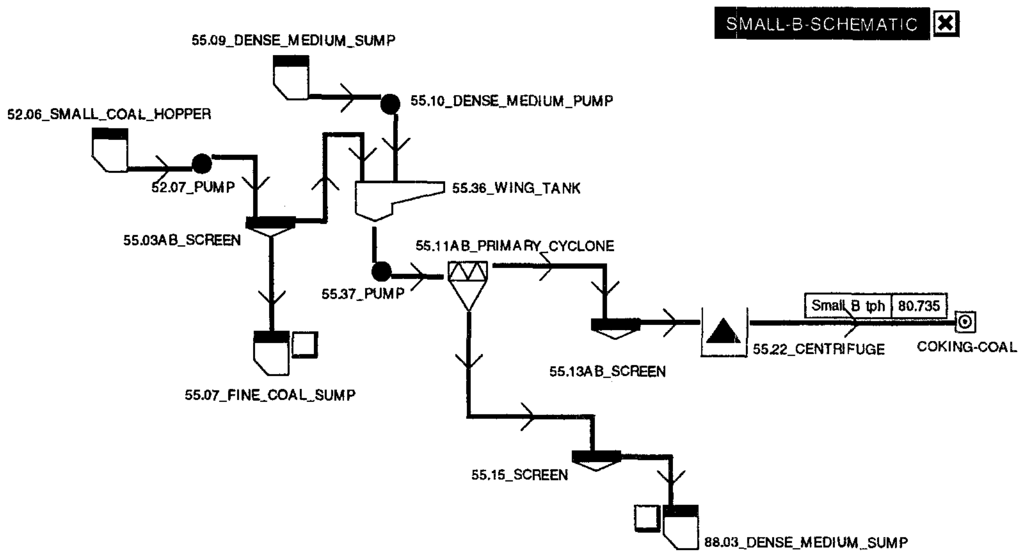


Fig. 5. Plant B small coal circuit schematic.

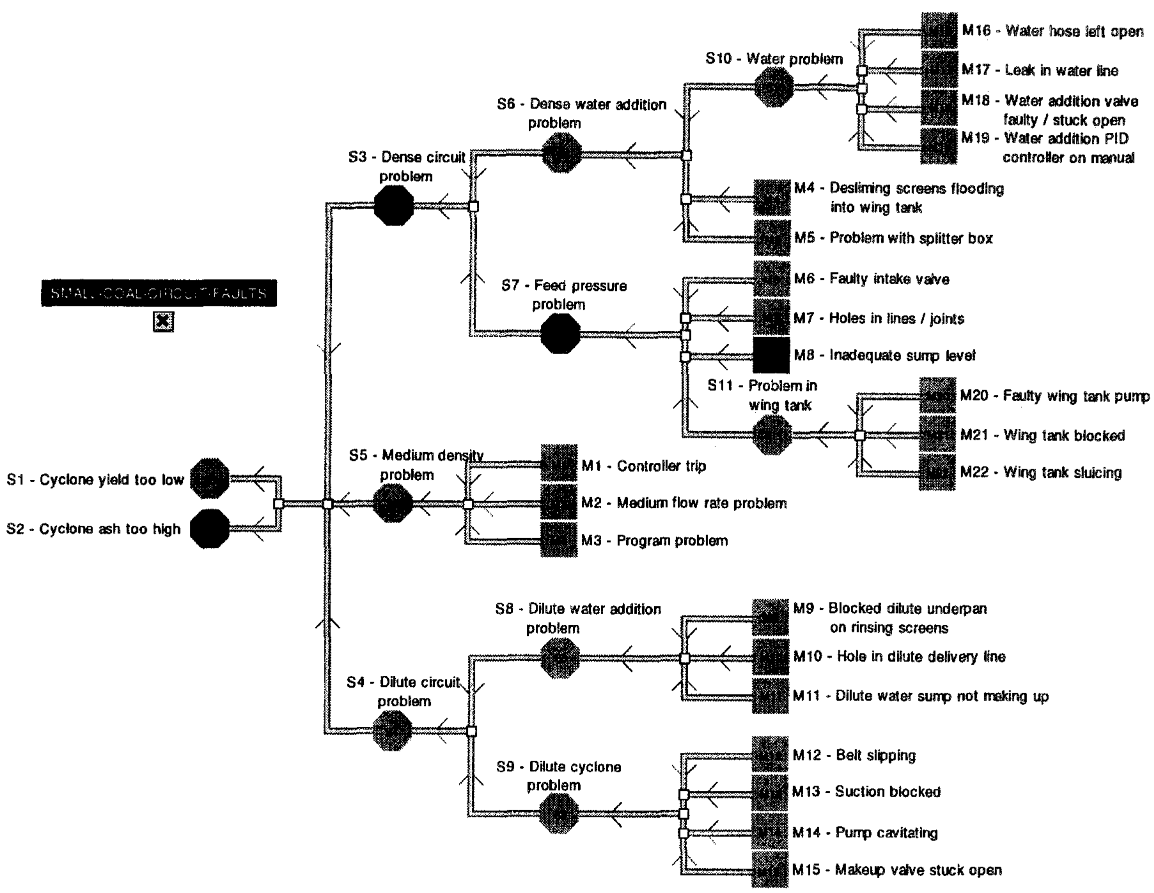


Fig. 6. Fault cause network for the small coal circuit.

In AshMod, SPC criteria are used to determine the success or failure of the ST goals. The state of the ST goals is then propagated upwards for the evaluation of the state of the GT goals. The value of each goal, and its animated colour (red, orange or green) is determined on the basis of how well the goal satisfies the SPC criteria.

When multiple ST goals fail the SPC criteria, the upward propagation may be a worst-case propagation or a weighted propagation. In worst-case propagation, the state of the worst ST goal is propagated to the top GT goal to capture the operator's attention. In weighted propagation, a weighted average of the state of each ST goal is propagated upwards, so that the GT goals get assigned a value that reflects the 'average' state of the plant.

A major attraction of the GTST model is that the very process of constructing it assists the domain experts and the knowledge engineers in formalising knowledge.

Each low level goal in the success tree has a list of indicators, instruments, parameters, operator activities, events and calculated parameters that effect the success or faliure of the goal. For example:

Goal No.	Goal Statement	Sensor No.
55	Prevent overheating of heater rods	70

Associated with each indicator[3] is a range of values that represents the acceptable range for that indicator. A similar indication of acceptability is used for calculated parameters, events, operator activities etc. For example:

Sensor No.	Instrument	Unit	Lower bound	Upper bound
70	Thermocouple	°C	80	100

Further, for each instrument gathering plant data, its location in the plant is established. For example:

Sensor No.	Instrument	Location
70	Thermocouple	Coolant tower

G3 (Fig. 4) is an instance of the goal class. Associated with each goal is a goal value and a goal weight that serve to establish the state of each goal. The G2 knowledge base is modularised into workspaces. The workspace shown in Fig. 2 graphically captures the relationship between goals and subgoals. Black squares represent individual goals, and the directed links are the 'connections' between goals. These directed links may be used at startup to infer named 'relations' between goals. The following initialisation rule is fired at startup and sets up the 'supergoal-of' relation between goals: *Initially for any goal g1 connected at an input of any goal g2 unconditionally conclude that g1 is*

supergoal-of g2. The state of a supergoal is determined through the recusrive propagation of the state of its subgoals. The following G2 rules are used for state propagation:

(1) *For any goal g1 if there exists a goal that is subgoal-of g1 and the value of propagation strategy is proactive then conclude that the goal-value of g1 = the maximum over each goal g2 that is subgoal-of g1 of (the goal-value of g2).*
(2) *For any goal g1 if there exists a goal that is subgoal-of g1 and the value of propagation strategy is reactive then conclude that the goal-value of g1 = the sum over each goal g2 that is subgoal-of g1 of (the goal-value of g2 * the goal-weight of g2).*

6 PLANT SCHEMATIC

The Plant Schematic is a KADS domain knowledge model that captures the structure of the plant, its components, their connectivity, and the flow of information, control and material.[3] In the AshMod schematic (Fig. 5), the components are not labelled so as to avoid visual cluster since standard icons are used to represent each component. A high level schematic provides an overview of the plant. Numerous low level schematics allow the operator to 'drill down' to detailed representations of individual circuits (plant components). Through the use of schematics, an operator guidance system can 'understand' that any two connected components can influence each other. For instance, a problem observed in one component could be due to a malfunction in a neighbouring component further upstream or downstream. AshMod understands that a 'water addition problem' observed in the primary cyclone could be due to a 'flooding' malfunction in the desliming screens (upstream neighbour of the primary cyclone) or due to a 'blockage' malfunction in the secondary cyclone or the basket centrifuge (downstream neighbours of the primary cyclone). This information cannot be obtained from any other knowledge model but is clearly captured by the schematic.

In addition, the plant schematic provides a dynamically animated display that shows the state of the entire plant as well as the state of the components of the plant. When AshMod determines that the jig is operating out of statistical process control, its schematic icon is highlighted to attract the operator's attention. The operator may then click on the jig's icon to investigate the problem through a fault–cause diagnosis performed by AshMod.

7 FAULT CAUSE NETWORK

The Fault Cause Network is a KADS domain knowledge model that captures plant behaviour and related operator heuristics. Operators use heuristic models based on first principles to perform diagnosis. For the sake of speed, efficiency and timeliness, AshMod makes use of compiled heuristic knowledge in the form of a fault cause network (Fig. 6). This network provides a means of identifying the root cause of an observed fault in the plant. The octagons in the figure represent faults, while the squares represent root causes. Hence low cyclone yield (S1) could be because of S3, S4 or S5. AshMod traces back from an observed fault (say S1) through intermediate faults (S3, S6 and S10) to identify the root cause (say M16).

Faults (for example S6 — dense water addition problem) are the symptoms or problems that are discovered by the operator or by the operator guidance system. Causes (for example, M17 — leak in water line) are the culprits or malfunctions that generate the fault. The root cause of a fault must be identified and fixed to remove the fault.

By integrating Bayes' theorem into the fault cause network, AshMod uses probabilistic reasoning to determine its confidence in a hypothesis. This determines the probability that a certain malfunction is the root cause, given an observed fault. This allows AshMod to provide a level of confidence with every recommendation. As a result the operator is given the freedom to use his own judgement in deciding upon a course of action. Operator feedback is used to adjust these confidence factors through reinforced learning. When the operator accepts or rejects a recommendation, the confidence of the potential causes associated with the recommendation is increased or decreased accordingly. Consider a fault F and its two possible causes C1 and C2. Assume that F is caused 40% of the time by

S5, a fault	
Notes	OK
Item configuration	none
Names	S5
Statement	"Medium density problem"
Status	unknown
State	active
Confidence	35.508

M16, a cause	
Notes	OK
Item configuration	none
Names	M16
Statement	"Water hose left open"
Status	unknown
State	inactive
Confidence	20
Action	"Turn it off and inform Rob Sedmak"

Fig. 7. Table for the small coal circuit S5 and cause M16.

C1, and 60% of the time by C2. Every time C1 is found to be the culprit, its probability is increased by a very samll amount, and C2's probability decreased, in such a way that their total probability continues to be 100%. Since the changes are very small, over time the increases and decreases cancel each other out, and the probabilities remain at approximately 40% and 60% respectively. However, if, due to gradual changes in plant behaviour, the actual probabilities are closer to 45% and 55% respectively, then, over time, the small adjustments will move the recorded probabilities toward the actual values.

S5 and M16 (Fig. 7) are instances of the fault and cause classes. Associated with each fault and/or cause is a confidence that serves to establish the most likely cause of an observed fault. The following G2 rules are used to highlight in red all observed faults and causes, and to highlight in blue the most likely cause of an observed fault:

(1) *For any occurrence O whenever the status of O receives a value and when the status of O is alarm then change the back icon-color of O to red.*
(2) *For any occurrence O1 that is caused-by any occurrence O2 whenever the status of O1 is alarm and for every occurrence O3 that is the-cause-of O1 (the confidence of O2 ≥ the confidence of O3) then change the back icon-color of O2 to blue.*

8 MATHEMATICAL MODELS

Mathematical models are based on observations of behavioural relationships between plant components. AshMod uses numerous mathematical models, a couple of which are mentioned here.

Based on simple physical principles, the fine coal tons per hour is linearly proportional to the instantaneous volume of fine coal on the conveyor belt. This, in turn, is linearly proportional to the instantaneous conveyor speed–depth product. The constant of proportionality is estimated by making actual measurements of the conveyor tons per hour and comparing them with the recorded conveyor speed and depth. By applying linear regression on numerous such measurements, the following linear model was constructed: fine coal tons per hour = 0.025 × (conveyor belt speed × depth of fine coal on conveyor belt). An occasional verification is required to test the accuracy of the constant of proportionality.

Coal washing researchers use the following model to capture the relationship between target ash and target yield:[3] $R_c = Y_p \times ((100 - P_A)/(100 - F_A))$, where R_c = carbon recovery, Y_p = target yield, P_A = target ash, F_A = feed ash. The carbon recovery is assumed constant, depending on the coal washing process being used (jig, cyclone, or flotation). Feed ash is estimated from feed coal quality. This model is used by AshMod to estimate unknown product ash values in the different circuits within the plant.

FINAL-ASH, a gfi-quantitative-variable	
Options	do not forward chain, breadth first backward chain
Notes	OK
Item configuration	none
Names	FINAL-ASH
Tracing and breakpoints	default
Data type	quantity
Initial value	9.0
Last recorded value	7.5, valid indefinitely
History keeping spec	keep history with maximum number of data points = 100
Validity interval	indefinite
Formula	none
Simulation details	no simulation formula yet
Initial value for simulation	default
Data server	GFI data server
Default update interval	none
GFI input interface object	data-file
Aim	9.0
Stdev	2.0
Shorttermtrend	increasing
Midtermtrend	increasing
Longtermtrend	decreasing
Trend	steady
Validity	valid
Validated value	7.5
Callibrated value	7.5
Filtered value	7.335
Level	normal
Filter coeff	0.95

Fig. 8. Process variable Final-Ash.

9 MONITORING

KADS provides a library of generic interpretation models for the monitoring task. These were instantiated and refined for this task in the coal washing domain. AshMod employs a data driven or forward chaining strategy with the monitoring task.[3] The results of the monitoring task are immediately made available to the diagnosis and optimisation tasks. Final-Ash (Fig. 8) is an instance of the process variable class. As each process variable is monitored, the associated information is maintained in its object table. The monitored process variable trends are displayed to the operator (Fig. 9).

9.1 Validation

To identify sensor failure, AshMod validates the raw data. Since the software is currently undergoing online testing, a

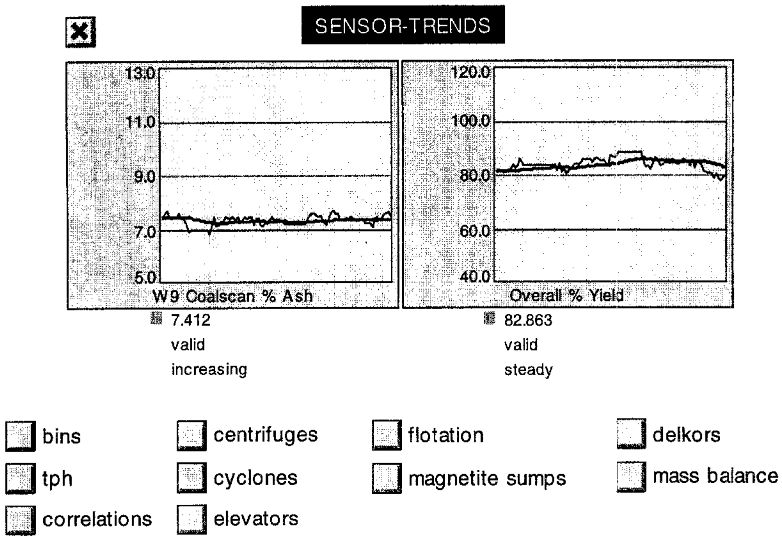


Fig. 9. The trends workspace.

direct data feed is provided to the operator guidance system from a centralised data repository, via a serial link. The following G2 rules are used for validation:

(1) *For any gfi-quantitative-variable v if abs(the current value of v − the aim of v) ≤ 3 * the stdev of v then conclude that the validated-value of v = the current value of v and conclude that the validity of v = the symbol valid.*

(2) *For any gfi-quantitative-variable v if abs(the current value of v − the aim of v) > 3 * the stdev of v then conclude that the validated-value of v = the aim of v and conclude that the validity of v = the symbol invalid.*

9.2 Filtration

AshMod uses the exponentially weighted moving average (EWMA) filter, which is a good filter in situations where the process mean moves slowly relative to the movements caused by measurement noise.[3] A benefit of using this filter is that it has an incremental formulation, so that the computation time required for real-time filtering is not excessive. The current value of the process variable depends, in an exponentially weighted manner, on all the prior values of the variable: $x_k = w \times x_k + (1 - w) \times x_{k-1}$, where $0 < w < 1$. As with many filters, the EWMA filter introduces a delay that is a function of the filter coefficient w. This coefficient can be adjusted by the operator to obtain a satisfactory tradeoff between delay and sensitivity to noise spikes.

9.3 Calibration

Poorly calibrated sensors produce signal values that have a constant offset. Calibration of process sensors is not undertaken very often due to safety and economic considerations. Hence, to improve the reliability of future analysis, AshMod performs automatic calibration of all sensor data. It averages out signal variability over a long time window. If the sensor is correctly calibrated, this average would be the same as the target value of the process variable. Any deviation from this equality suggests a calibration error, which is then corrected. The following G2 rule is used for calibration: *for any variable v unconditionally conclude that the calibrated-value of v = the validated-value of v − (the avg value of v during the last 6 hours − the aim of v).*

9.4 Trending

As shown in Fig. 9, a short-term, mid-term and long-term trend is computed for each process variable. This is based on the comparison of the current filtered value and the filtered value 1, 2 and 4 hours ago. If all three trends are increasing (or decreasing), then the overall trend is assumed to be increasing (or decreasing). Otherwise the overall trend is assumed to be steady.

9.5 Statistical process control

AshMod applies statistical process control criteria on each subsystem in the process. To do so, it examines the current as well as historical values and trends of all process variables. When the subsystem fails to satisfy the SPC criteria, the diagnosis task is initiated to identify, diagnose and remove faults. The G2 procedure illustrated in Fig. 10 is used to determine the state of the goal g associated with process variable pv.

10 DIAGNOSIS

KADS provides a library of generic interpretation models for the diagnosis task. These were instantiated and refined for this task in the coal washing domain. AshMod performs diagnosis through causal tracing. A fault cause network is used to determine the root cause of an observed fault.[3] AshMod identifies the components suspected to contain faults, the known and suspected faults (Fig. 11), the root

SPC, a procedure	
Notes	OK
Authors	lamba@ harmeet @ hs (4 Nov 1996 8:53 a.m.)
Item configuration	none
Tracing and breakpoints	default
Class of procedure invocation	none
Default procedure priority	6
Uninterrupted procedure execution limit	use default

```
spc (pv : class gfi-quantitative-variable, g: class goal)
i, outOfControl, normalState: integer;
begin

  for i = 0 to 7 do
    conclude that spc-array [i] = the value of pv as of I datapoints ago;
  end;

{ more than 1/8 outside 3 SD implies out-of-control }

  if the count of each float f in spc-array such that (f < the aim of pv - 3 * the stdev
     of pv) > 1
  then begin
    conclude that the goal-value of g = outOfControl;
    return
  end;

  if the count of each float f in spc-array such that (f > the aim of pv + 3 * the stdev
     of pv) > 1
  then begin
    conclude that the goal-value of g = outOfControl;
    return
  end;

{ 8/8 outside 1 SD implies out-of-control }

  if the count of each float f in spc-array such that (f > the aim of pv + the stdev of
     pv) = 8
  then begin
    conclude that the goal-value of g = outOfControl;
    return
  end;

  if the count of each float f in spc-array such that (f < the aim of pv - the stdev of
     pv) = 8
  then begin
    conclude that the goal-value of g = outOfControl;
    return
  end;

{ else assume normal state}

  conclude that the goal-value of g = normalState;

end
```

Fig. 10. Statistical process control procedure.

causes or malfunctions that caused the faults and the corrective action needed to remove the fault. When an operator observes a goal tree node in 'alarm' state, he concludes that a particular circuit or component in the plant is out of statistical process control, and hence initiates the diagnosis task. This directs the operator guidance system to identify the malfunction(s) that have caused the out-of-control state. AshMod automatically focuses diagnosis on the appropriate plant circuit. Hence the essential strategy employed with the diagnosis task is event driven.

10.1 Detection

AshMod maintains a list of potential faults that can be detected by the system or by the operator through the use of available process trends and states. Hence, a fault detection procedure is specified for each fault. Where AshMod has the knowledge and information to use the procedure, it does so; otherwise it uses the procedure to guide the operator so as to assist him in identifying the fault. For example, a fault S1 is detected using the following F2 rule: *if the level of smallb is high and the trend of smallb is increasing then conclude that the status of S1 is alarm.*

10.2 Diagnosis

Causal tracing uses the fault cause network to progress from an observed symptom through intermediate faults and causes till the root cause is identified. At each step, AshMod presents a list of potential faults and possible malfunctions to the operator, together with confidence factors to indicate the probability that the listed fault or cause is responsible. The operator may accept or reject these suggestions, thus allowing AshMod to adjust its confidence factors through the application of reinforced learning principles.

10.3 Removal

Once the root cause of a fault has been determined, AshMod recommends an appropriate corrective action to the operator. Due to the time constants associated with plant components, it takes time before an operator action has an observable effect on the plant behaviour. During this period the same action recommendation should not be repeated. AshMod knows the 'reaction time' for each action, and hence does not make the same recommendation till the plant has been given enough time to respond to the operator action.

11 OPTIMISATION

In most production environments, considerable gains can be achieved through optimisation.[3] At the coal washery, optimisation is seen along two dimensions.

First, the product ash must be kept within a specified range. The ash provides a measure of the amount of impurity in the product coal. A high ash content reduces the quality of the coal product and hence reduces its value. The plant personnel have a target ash value in mind, and the operation of the plant is tuned to try to achieve that target. It is possible to mix a high ash product with a low ash product to achieve the target ash, but this is generally accompanied by lost yield. Hence, to maximise the yield, it is desirable to operate all the plant circuits at the target ash all the time. The larger the product ash variation between circuits and over time, the greater the loss in product yield.

Second, assuming that the product ash is on target, the product yield should be maximised. This optimisation objective is in fact indirectly satisfied by the fulfilment of the first objective. If all plant circuits are producing ash at target, then the yield will automatically be maximised. This is because of the inherent trade-off between ash and yield. An increase in yield comes at the cost of increased ash content. A decrease in the ash content comes at the cost of lost yield. However, the relationship is not linear. Operating

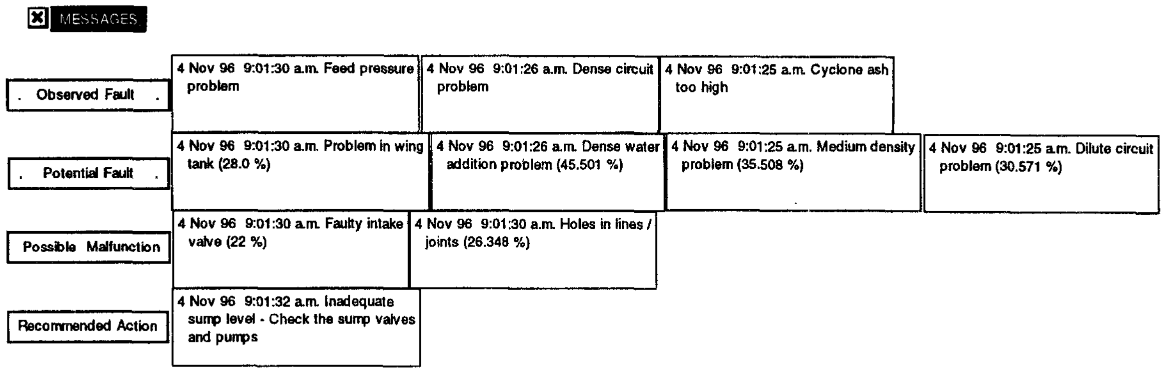


Fig. 11. AshMod's diagnosis messages workspace.

at the knee of this curve allows the plant to achieve the optimum balance between ash and yield.

To achieve this objective, AshMod must first determine the optimum operating point of the plant. This is determined on the basis of the input raw coal feed quality. Thereafer, the optima for each of the six circuits (the large, small and fine coal circuits of plants B and C) must be evaluated. Finally, an optimisation strategy is determined and presented to the operator (Fig. 12). This strategy identifies the circuit whose setpoint the operator should modify.

The optimisation function has only one decision variable. Achieving the ash aim automatically results in maximum yield. The plant ash is a function of the yield and vice versa. A similar relationship exists for each circuit within the plant. The plant optimum must lie on the target ash line (Fig. 13). The actual position of the optimum depends on the feed quality.

The following G2 functions are used to determine the ash and yield target for each circuit:

(1) *circuitYield(productTPH, feedProportionrawTPH) = productTPH/(feedProportion * rawTPH)*

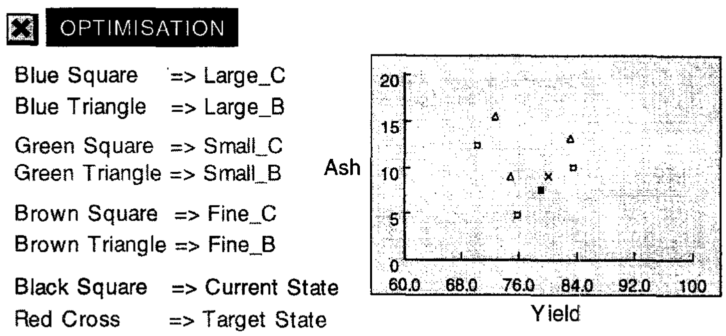
(2) *circuitAsh(circuitYield, feedAsh, carbonRecovery) = 100 – (((100 – feedAsh) * carbonRecovery)/circuitYield)*

The determination of the circuit target states is essential, because each of the individual circuits must be optimised to optimise the entire plant. The current state of each circuit is compared with its goal state to evaluate its performance. The circuits that are performing poorly are the suitable candidates for optimisation.

The optimisation strategy synchronises the twin circuits (eg Jig_B and Jig_C). For optimum operation both plants must operate in synchronisation (at the same operating point). Once all circuits are synchronised, they are jointly pushed towards the target. For each circuit an optimality coefficient is defined by the expression (| target yield – current yield |)/(target yield). The circuit with the largest optimality coefficient is optimised first, since it provides maximum potential for improvement.

12 CONCLUSIONS

AshMod bridges the gap between theory and practice by



Circuit	Optimality Coefficient	Ash Action	Yield Action
Large B	0.252	decrease	decrease
Large C	0.248	increase	increase
Small B	0.412	decrease	decrease
Small C	0.248	increase	increase
Fine B	0.094	increase	increase
Fine C	0.08	decrease	decrease

Fig. 12. AshMod's optimisation workspace.

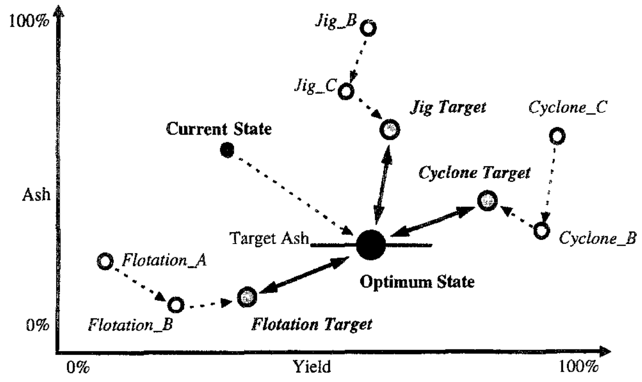


Fig. 13. The sequence of steps that take the plant from current state to optimum state.

applying the sound knowledge modelling principles propounded by the KADS methodology in an operator guidance system that will operate in an industrial environment. KADS has provided us with a systematic framework for the development of knowledge models, thus significantly simplifying knowledge acquisition and the transition to design and implementation.

We have presented in this paper a metaphor for capturing expert knowledge through knowledge models that correspond with the way experts view their domain. There are multiple perspectives from which a domain expert would view a plant. These are best captured through topological, geographical, behavioural, means–end, mathematical and task models.

The knowledge engineer has to extract domain, inference, task and strategy knowledge from these models to construct the KADS expertise models, which correspond with the way a knowledge engineer needs to view the domain to construct a software solution. The expertise models are then mapped to the design models and implemented in software.

AshMod is currently undergoing testing and validation. In offline testing, when a process upset occurs on a particular day, data from that day is fed into AshMod to determine whether any warnings are generated. On numerous occasions, it has been found that the operator guidance system could have advised the operator 15–20 minutes prior to the occurrence of a major plant upset, hence potentially preventing shutdown. We have recently placed AshMod online, to observe what useful and false positive advice is generated as live data is fed into the system, 24 hours a day, 7 days a week.

AshMod currently recieves data values for 60 process variables (obtained from hard sensors on the plant), and models another 20 process variables (soft sensor models, derived from the available data). New sensors will be installed and more process variables made available to AshMod as the operator guidance system starts paying dividends during online operation. The availability of additional sensor data will further improve the quality of recommendations made by the system. In certain situations, AshMod is able to provide only a high level fault diagnosis, where the root cause could be one of numerous potential candidates. Operator input is required to identify the precise culprit. This could be avoided if more low level sensor data was available to the operator guidance system. The robustness of the system and the reliability of its recommendations will also increase with the addition of new and redundant process variables.

On the basis of current performance, it is estimated that, once completely operational, AshMod could generate savings of up to a million dollars (Australian) per annum. With a total development cost of a quarter of a million dollars, we expect a payback time of approximately 3 months.

ACKNOWLEDGEMENTS

This work was supported by BHP and the Centre for Industrial Control Science at the University of Newcastle, Australia. We thank Gensym for providing excellent support for G2, John Ellison and Rob Sedmak at BHP FPD for being patient in awaiting the results of this effort, and the anonymous referee for providing excellent feedback on the initial version of this paper.

REFERENCES

1. Broken Hill Proprietary Company Limited Home Page: http://www/bhp.com.au/.
2. BHP, *BHP in Perspective*. 1996.
3. Lamba, H., Knowledge based operator guidance system. Department of Electrical and Computer Engineering, The University of Newcastle, Australia, 1995.
4. Coal Preparation Society of New South Wales, *An Introduction to Coal Preparation*. Port Kembla, Australia, 1990.
5. BHP Coal Preparation Department, *Washery Training Manual*. Port Kembla, Australia, 1994.
6. Hickman, F., Killin, J., Land, L., Mulhall, T., Porter, D. & Taylor, R., *Analysis for Knowledge-based Systems: A Practical Guide to the KADS Methodology*. Ellis Horwood, West Sussex, 1989.
7. Schrieber, G., Wielinga, B., de Hoog, R., Akkermans, H. & Van de Velde, W., CommonKADS: a comprehensive methodology for KBS development. *IEEE Expert*, Vol. 9, No. 6, 1994.
8. Gensym Corporation, *An Introduction to G2*. Cambridge, MA, 1990.
9. Gensym Corporation, *G2: A Strategic Choice for Building Intelligent Real-time Applications*. Cambridge, MA, 1995.
10. Kim & Modarres, Application of goal tree–success tree models as the knowledge-base of operator advisory systems. *Nuclear Engineering and Design*, 1987, **104**, 67–81.
11. Villanueva, H. & Arena, S., HeatX: Expert system thermal performance analysis in power plant operation. *Eighth Australian Joint Conference on Artificial Intelligence*, Canberra, Australia, 1995.
12. Villanueva, H. & Arena, S., Automatic object creation and maintenance in G2 knowledge bases. *Gensym's World User Conference*, Boston, 1995.
13. Villanueva, H. & Arena, S., Automation of object creation and maintenance in object oriented systems. *ICX 95*, Sydney, Australia, 1995.

14. Villanueva, H. & Lamba, H., AshMod: Knowledge model based operator guidance system for coal washing. *AIENG 96*, Florida, 1996.

15. Villanueva, H. & Lamba, H., Operator guidance system for industrial plant supervision. *Expert Systems with Applications*, 1997, **12**(4), 441–454.